UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/688,573	10/20/2003	Robert M. Zeidman	ZEID-01	2483

66323 7590 09/01/2010

ZEIDMAN TECHNOLOGIES, INC.
15565 SWISS CREEK LANE
CUPERTINO, CA 95014

EXAMINER	
WANG, BEN C	

ART UNIT	PAPER NUMBER
2192	

MAIL DATE	DELIVERY MODE
09/01/2010	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	Application No.	Applicant(s)	
	10/688,573	ZEIDMAN, ROBERT M.	
	Examiner	Art Unit	
	BEN C. WANG	2192	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *14 June 2010*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1,3-7,15,17-22 and 24-34* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1, 3-7, 15, 17-22, and 24-34* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

1. Applicant's amendment dated June 14, 2010, responding to the April 14, 2010 Office action provided in the rejection of claims 1, 3-7, 15, 17-22, and 24-34, wherein claims 1, 15, 22 and 29 have been amended.

Claims 1, 3-7, 15, 17-22, and 24-34 remain pending in the application and which have been fully considered by the examiner.

Applicant's arguments with respect to claims currently amended have been fully considered but are moot in view of the new grounds of rejection – see *Mathur et al.* and *Yodaiken* - arts made of record, as applied hereto.

2. Applicant's amendment necessitated the new ground(s) of rejection presented in this Office action. Accordingly, **THIS ACTION IS MADE FINAL**. See MPEP § 706.07(a).

Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory

action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the date of this final action.

Claim Rejections – 35 USC § 103(a)

The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made

3. Claims 1, 3, 5, 7, 15, 17, 19, 21-22, 24, 26, 28-30, 32, and 34 are rejected under 35 U.S.C. 103(a) as being unpatentable over Lehman et al. (US Patent 4,796,179) (hereinafter 'Lehman') in view of Mathur et al. (US Patent 6,671,745 B1) (hereinafter 'Mathur' - art made of record)

4. **As to claim 1** (Currently Amended), Lehman discloses a method for developing a real-time operating system (e.g., Fig. 1; Col. 1, Lines 46-48; Col. 4, Lines 63-68; Col. 5, Lines 1-2), comprising:

specifying a set of *n* tasks (e.g., Col. 1, Lines 33-38), task(1) through task(n), to be scheduled for execution; (e.g., Abstract, Lines 8-14; Col. 3, Lines 1-8; Col. 5, Lines 5-12; Col. 135, Lines 17-24).

Further, Lehman discloses generating computer program for controlling real time process (e.g., Col. 1, Lines 5-8 – emphasis added) and at least one of

the task of the set of *n* tasks being selected as a preemptive or a non-preemptive task (e.g., Col. 9, Lines 52-56; Col. 10, Lines 3-12; Col. 35, Lines 1-5; Col. 136, Lines 40-50); synthesizing source code to implement a task scheduler (e.g., Fig. 4, SCHEDULER 24; Fig. 24 – SCHEDULER FLOW CHART ; Fig. 26; Col. 2, Lines 36-39; Col. 3, Lines 36-39; Col. 9, Lines 56-61; Abstract, Lines 20-25; Col. 10, Lines 8-12 – emphasis added) but Lehman does not explicitly disclose other limitations stated below.

However, in an analogous art of *Application Program Interfaces and Structures in a Resource Limited Operating System*, Mathur discloses:

specifying t init-tasks that are executed only once upon initial execution of a task scheduler, t being less than or equal to n (e.g., Col. 9, Lines 48-67 - … performs the following tasks: initiates the loading of a driver at system start up …; Col. 9, Lines 14-40 - … the OAL module 208 includes interfaces … Interrupt Service Routine (ISR) handlers to support device deriver; real-time clock (RTC); Interval timer (used for the scheduler operation); **NOTE**: Lehman teaches a task scheduler as previously stated; In addition, 'a task scheduler' in a real-time system is well-known in the art as cited in the section of Background of the Invention in the pending specification, e.g., Lines 13-20 on page 1 - … performs a number of response time-critical tasks according to a schedule … according to the scheduling algorithm ... – emphasis added);

using a data processor to synthesize source code from commands embedded in source code to implement the task scheduler for controlling execution of said set of n tasks (e.g., Fig. 3, MODULE API 302, COMPONENT

API 304; Col. 10, Lines 17-31 - … Module 308 exposes an API 302 to application … allows the application to interface and call methods or functions implemented by the module ... – emphasis added), the task scheduler further controlling one execution of each of said set of t init-tasks, said synthesized source code being executable on a target system after compilation (e.g., Col. 10, Lines 32-44 - … one benefit is that the operating system is modular … allows an embedded system designer to create an operating environment that is optimized for their unique hardware development platform and application … - emphasis added).

synthesizing source code from commands embedded in source code to control execution of said set of t init-tasks (e.g., Fig. 2, OAL 208; Col. 9, Lines 14-40 - … the OEM Adaption Layer (OAL) module 208 allows an embedded system developer to adapt the operating system for a specific target platform … the OAL module 208 includes interfaces … Interrupt Service Routine (ISR) handlers to support device deriver; real-time clock (RTC); Interval timer (used for the scheduler operation) - emphasis added).

Therefore, it would have been obvious to one of ordinary skill in the pertinent art, at the time the invention was made to combine the teachings of Mathur into the Lehman's system to further provide other limitations stated above in the Lehman system.

The motivation is that it would further enhance the Lehman's system by taking, advancing and/or incorporating the Mathur's system which offers significant advantages that the operating system should provide APIs for components and modules that meet the unique input and output needs of an

embedded system as once suggested by Mathur (e.g., Col. 2, Lines 36-49 – emphasis added)

5. **As to claim 3** (Previously Presented) (incorporating the rejection in claim 1), Lehman discloses the method and the apparatus further including specifying *f f-loop* tasks, each having an associated integer value *li* for *i* ranging from *1* to *f* and *f* being less than or equal to *n* (e.g., Col. 20, line 63 through Col. 21, line 20 – for loops using an incrementing or decrementing counter, i.e. Loop for *I* = 1 to *X* (executing) block of statements), said task scheduler addresses the task scheduler executing the loops including a continuously executing loop such that each *f-loop* task executes exactly once every *li* times that the loop is executed (e.g., Col. 21, Lines 13-19)

6. **As to claim 5** (Previously Presented) (incorporating the rejection in claim 1), Lehman discloses the method and apparatus further including specifying *c call-tasks*, *c* being less than or equal to *n*, the task scheduler scheduling a *call-task* when another task requests that the *call-task* be executed (e.g., Col. 7, Lines 29-32; Col. 16, Lines 21-23)

7. **As to claim 7** (Previously Presented) (incorporating the rejection in claim 1), Lehman discloses the method and the apparatus where tasks are given priority values such that whenever the task scheduler chooses between scheduling multiple tasks, all of which being ready to be executed, said task

scheduler chooses from among those tasks that have the highest priority values (e.g., Col. 2, Lines 36-39; Col. 9, Lines 62-68; Col. 10, Lines 1-2; Col. 32, Lines 5-54)

8. **As to claim 15** (Currently Amended), Lehman discloses an apparatus for developing a real-time operating system comprising:

a computer (e.g., Fig. 27 – multi-process controller, Lines 30-32);

a computer readable medium in data communication with the computer (e.g., Col. 135, Line 15 through Col. 137, Lines 64), the computer readable medium including a software synthesis program stored thereon (e.g., Col. 135, Line 15 through Col. 137, Lines 64), which when executed by the computer causes the computer to specify a set of *n* tasks (e.g., Col. 1, Lines 33-38), task(1) through task(n), to be scheduled for execution; (e.g., Abstract, Lines 8-14; Col. 3, Lines 1-8; Col. 5, Lines 5-12; Col. 135, Lines 17-24)

Further, Lehman discloses generating computer program for controlling real time process (e.g., Col. 1, Lines 5-8 – emphasis added) and at least one of the task of the set of *n* tasks being selected as a preemptive or a non-preemptive task (e.g., Col. 9, Lines 52-56; Col. 10, Lines 3-12; Col. 35, Lines 1-5; Col. 136, Lines 40-50); synthesizing source code to implement a task scheduler (e.g., Fig. 4, SCHEDULER 24; Fig. 24 – SCHEDULER FLOW CHART ; Fig. 26; Col. 2, Lines 36-39; Col. 3, Lines 36-39; Col. 9, Lines 56-61; Abstract, Lines 20-25; Col. 10, Lines 8-12 – emphasis added) but Lehman does not explicitly disclose other limitations stated below.

However, in an analogous art of *Application Program Interfaces and Structures in a Resource Limited Operating System*, Mathur discloses:

specifying t init-tasks that are executed only once upon initial execution of a task scheduler, t being less than or equal to n (e.g., Col. 9, Lines 48-67 - … performs the following tasks: initiates the loading of a driver at system start up …; Col. 9, Lines 14-40 - … the OAL module 208 includes interfaces … Interrupt Service Routine (ISR) handlers to support device deriver; real-time clock (RTC); Interval timer (used for the scheduler operation); **NOTE**: Lehman teaches a task scheduler as previously stated; In addition, 'a task scheduler' in a real-time system is well-known in the art as cited in the section of Background of the Invention in the pending specification, e.g., Lines 13-20 on page 1 - … performs a number of response time-critical tasks according to a schedule … according to the scheduling algorithm ... – emphasis added);

synthesizing source code from commands embedded in source code to implement the task scheduler for controlling execution of said set of n tasks(e.g., Fig. 3, MODULE API 302, COMPONENT API 304; Col. 10, Lines 17-31 - … Module 308 exposes an API 302 to application … allows the application to interface and call methods or functions implemented by the module ... – emphasis added), said synthesized source code being executable on a target system after compilation (e.g., Col. 10, Lines 32-44 - … one benefit is that the operating system is modular … allows an embedded system designer to create an operating environment that is optimized for their unique hardware development platform and application … - emphasis added);

synthesizing source code from commands embedded in source code to control execution for said set of t init-tasks (e.g., Fig. 2, OAL 208; Col. 9, Lines 14-40 - ... the OEM Adaption Layer (OAL) module 208 allows an embedded system developer to adapt the operating system for a specific target platform ... the OAL module 208 includes interfaces ... Interrupt Service Routine (ISR) handlers to support device deriver; real-time clock (RTC); Interval timer (used for the scheduler operation) - emphasis added)

Therefore, it would have been obvious to one of ordinary skill in the pertinent art, at the time the invention was made to combine the teachings of Mathur into the Lehman's system to further provide other limitations stated above in the Lehman system.

The motivation is that it would further enhance the Lehman's system by taking, advancing and/or incorporating the Mathur's system which offers significant advantages that the operating system should provide APIs for components and modules that meet the unique input and output needs of an embedded system as once suggested by Mathur (e.g., Col. 2, Lines 36-49 – emphasis added)

9. **As to claim 17** (Previously Presented) (incorporating the rejection in claim 15), Lehman discloses the apparatus being further configured to specify *f f-loop* tasks, each having an associated integer value *I(i)* for *i* ranging from *1* to *f* and *f* being less than or equal to *n* (e.g., Col. 20, line 63 through Col. 21, line 20 – for loops using an incrementing or decrementing counter, i.e. Loop for *I* = 1 to *X*

(executing) block of statements), the task scheduler including a continuously executing loop such that each *f-loop* task executes exactly once every *l(i)* times that the loop is executed (e.g., Col. 21, Lines 13-19)

10. **As to claim 19** (Previously Presented) (incorporating the rejection in claim 15), please refer to above claim **5** accordingly.

11. **As to claim 21** (Previously Presented) (incorporating the rejection in claim 15), please refer to above claim **7** accordingly.

12. **As to claim 22** (Currently Amended), Lehman discloses an apparatus for developing a real-time operating system (e.g., Fig. 1; Col. 1, Lines 46-48; Col. 4, Lines 63-68; Col. 5, Lines 1-2) comprising:

a computer;

a computer readable medium in data communication with the computer, the computer readable medium including a software synthesis program stored thereon, the software synthesis program including:

means for specifying a set of n tasks (e.g., Col. 1, Lines 33-38), task (1) through task (n), to be scheduled for execution (e.g., Abstract, Lines 8-14; Col. 3, Lines 1-8; Col. 5, Lines 5-12; Col. 135, Lines 17-24);

Further, Lehman discloses generating computer program for controlling real time process (e.g., Col. 1, Lines 5-8 – emphasis added) and at least one of the task of the set of *n* tasks being selected as a preemptive or a non-preemptive

task (e.g., Col. 9, Lines 52-56; Col. 10, Lines 3-12; Col. 35, Lines 1-5; Col. 136, Lines 40-50); synthesizing source code to implement a task scheduler (e.g., Fig. 4, SCHEDULER 24; Fig. 24 – SCHEDULER FLOW CHART ; Fig. 26; Col. 2, Lines 36-39; Col. 3, Lines 36-39; Col. 9, Lines 56-61; Abstract, Lines 20-25; Col. 10, Lines 8-12 – emphasis added) but Lehman does not explicitly disclose other limitations stated below.

However, in an analogous art of *Application Program Interfaces and Structures in a Resource Limited Operating System*, Mathur discloses:

means for specifying t init-tasks that are executed only once upon initial execution of a task scheduler, t being less than or equal to n (e.g., Col. 9, Lines 48-67 - … performs the following tasks: initiates the loading of a driver at system start up …; Col. 9, Lines 14-40 - … the OAL module 208 includes interfaces … Interrupt Service Routine (ISR) handlers to support device deriver; real-time clock (RTC); Interval timer (used for the scheduler operation); **NOTE**: Lehman teaches a task scheduler as previously stated; In addition, 'a task scheduler' in a real-time system is well-known in the art as cited in the section of Background of the Invention in the pending specification, e.g., Lines 13-20 on page 1 - … performs a number of response time-critical tasks according to a schedule … according to the scheduling algorithm ... – emphasis added);

means for synthesizing source code from commands embedded in source code to implement the task scheduler for controlling execution of said set of n tasks (e.g., Fig. 3, MODULE API 302, COMPONENT API 304; Col. 10, Lines 17-31 - … Module 308 exposes an API 302 to application … allows the application

to interface and call methods or functions implemented by the module ... – emphasis added), the task scheduler further controlling one execution of each of said set of t init-tasks (e.g.,), said synthesized source code being executable on a target system after compilation (e.g., Col. 10, Lines 32-44 - ... one benefit is that the operating system is modular ... allows an embedded system designer to create an operating environment that is optimized for their unique hardware development platform and application ... - emphasis added); and

means for synthesizing source code from commands embedded in source code to control execution of said set of t init-tasks (e.g., Fig. 2, OAL 208; Col. 9, Lines 14-40 - ... the OEM Adaption Layer (OAL) module 208 allows an embedded system developer to adapt the operating system for a specific target platform ... the OAL module 208 includes interfaces ... Interrupt Service Routine (ISR) handlers to support device deriver; real-time clock (RTC); Interval timer (used for the scheduler operation) - emphasis added)

Therefore, it would have been obvious to one of ordinary skill in the pertinent art, at the time the invention was made to combine the teachings of Mathur into the Lehman's system to further provide other limitations stated above in the Lehman system.

The motivation is that it would further enhance the Lehman's system by taking, advancing and/or incorporating the Mathur's system which offers significant advantages that the operating system should provide APIs for components and modules that meet the unique input and output needs of an

embedded system as once suggested by Mathur (e.g., Col. 2, Lines 36-49 – emphasis added)

13. **As to claim 24** (Previously Presented) (incorporating the rejection in claim 22), please refer to above claim **3** accordingly.

14. **As to claim 26** (Previously Presented) (incorporating the rejection in claim 22), please refer to above claim **5** accordingly.

15. **As to claim 28** (Previously Presented) (incorporating the rejection in claim 22), please refer to above claim **7** accordingly.

16. **As to claim 29** (Currently Amended), Lehman discloses a machine-readable medium embodying instructions which, when executed by a machine, cause the machine to:

specify a set of n tasks (e.g., Col. 1, Lines 33-38), task(l) through task(n), to be scheduled for execution (e.g., Abstract, Lines 8-14; Col. 3, Lines 1-8; Col. 5, Lines 5-12; Col. 135, Lines 17-24)

Further, Lehman discloses generating computer program for controlling real time process (e.g., Col. 1, Lines 5-8 – emphasis added) and at least one of the task of the set of *n* tasks being selected as a preemptive or a non-preemptive task (e.g., Col. 9, Lines 52-56; Col. 10, Lines 3-12; Col. 35, Lines 1-5; Col. 136, Lines 40-50); synthesizing source code to implement a task scheduler (e.g., Fig.

4, SCHEDULER 24; Fig. 24 – SCHEDULER FLOW CHART ; Fig. 26; Col. 2, Lines 36-39; Col. 3, Lines 36-39; Col. 9, Lines 56-61; Abstract, Lines 20-25; Col. 10, Lines 8-12 – emphasis added) but Lehman does not explicitly disclose other limitations stated below.

However, in an analogous art of *Application Program Interfaces and Structures in a Resource Limited Operating System*, Mathur discloses:

specifying t init-tasks that are executed only once upon initial execution of a task scheduler, t being less than or equal to n (e.g., Col. 9, Lines 48-67 - ... performs the following tasks: initiates the loading of a driver at system start up ...; Col. 9, Lines 14-40 - ... the OAL module 208 includes interfaces ... Interrupt Service Routine (ISR) handlers to support device deriver; real-time clock (RTC); Interval timer (used for the scheduler operation); **NOTE**: Lehman teaches a task scheduler as previously stated; In addition, 'a task scheduler' in a real-time system is well-known in the art as cited in the section of Background of the Invention in the pending specification, e.g., Lines 13-20 on page 1 - ... performs a number of response time-critical tasks according to a schedule ... according to the scheduling algorithm ... – emphasis added)

synthesizing source code from commands embedded in source code to implement the task scheduler for controlling execution of said set of n tasks (e.g., Fig. 3, MODULE API 302, COMPONENT API 304; Col. 10, Lines 17-31 - ... Module 308 exposes an API 302 to application ... allows the application to interface and call methods or functions implemented by the module ... – emphasis added), the task scheduler further controlling one execution of each of

said set of t init-tasks, said synthesized source being executable on a target system after compilation (e.g., Col. 10, Lines 32-44 - ... one benefit is that the operating system is modular ... allows an embedded system designer to create an operating environment that is optimized for their unique hardware development platform and application ... - emphasis added); and

synthesizing source code from commands embedded in source code to control execution of said set of t init-tasks (e.g., Fig. 2, OAL 208; Col. 9, Lines 14-40 - ... the OEM Adaption Layer (OAL) module 208 allows an embedded system developer to adapt the operating system for a specific target platform ... the OAL module 208 includes interfaces ... Interrupt Service Routine (ISR) handlers to support device deriver; real-time clock (RTC); Interval timer (used for the scheduler operation) - emphasis added)

Therefore, it would have been obvious to one of ordinary skill in the pertinent art, at the time the invention was made to combine the teachings of Mathur into the Lehman's system to further provide other limitations stated above in the Lehman system.

The motivation is that it would further enhance the Lehman's system by taking, advancing and/or incorporating the Mathur's system which offers significant advantages that the operating system should provide APIs for components and modules that meet the unique input and output needs of an embedded system as once suggested by Mathur (e.g., Col. 2, Lines 36-49 – emphasis added)

17. **As to claim 30** (Previously Presented) (incorporating the rejection in claim 29), please refer to above claim **3** accordingly.

18. **As to claim 32** (Previously Presented) (incorporating the rejection in claim 29), please refer to above claim **5** accordingly.

19. **As to claim 34** (Previously Presented) (incorporating the rejection in claim 29), please refer to above claim **7** accordingly.

20. Claims 4, 18, 25, and 31 are rejected under 35 U.S.C. 103(a) as being unpatentable over Lehman, in view of Mathur and Xu et al. *On Satisfying Timing Constraints in Hard-Real-Time Systems'*, 1991, ACM' (hereinafter 'Xu')

21. **As to claim 4** (Previously Presented) (incorporating the rejection in claim 1), Lehman discloses generating computer program for controlling real time process (e.g., Col. 1, Lines 5-8 – emphasis added) and at least one of the task of the set of *n* tasks being selected as a preemptive or a non-preemptive task (e.g., Col. 9, Lines 52-56; Col. 10, Lines 3-12; Col. 35, Lines 1-5; Col. 136, Lines 40-50)

But, Lehman and Mathur do not specifically disclose *p-loop* task.

However, in an analogous art, Xu discloses means for specifying *p-loop* tasks, each having an associated integer value *ti* for *i* ranging from *1* to *p* and *p* being less than or equal to *n*, the number *ti* representing a number of regular time units (e.g., Sec. 2, 3rd paragraph, Lines 1-4), said task scheduler including a

timer that schedules each *p-loop* task i to be executed approximately once every *ti* time units (e.g., Sec. 2, 3rd paragraph, Lines 1-4; Sec. 2, 7th paragraph, on page 133 – A periodic process *p* can be described by a quadruple(*rp, cp, dp, prdp*), where *prdp* is the period, *cp* is the worse case computation time required by process *p*, *dp* is the deadline, *rp* is the release time).

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the teachings of Xu into the Lehman-Mathur in order to provide a timing constraints mechanism in the Lehman-Mathur system.

The motivation is that (a) pre-run-time scheduling is essential if we want to guarantee that timing constraints will be satisfied in a complex hard-real-time system, (b) appropriate algorithms for solving mathematical scheduling problems that address those concerns can be used to automate pre-run-time scheduling, (c) if the task of computing schedules is completely automated, it would be very easy to modify the system and re-compute new schedules in case changes are required by applications as once suggested by Xu (e.g., Abstract, Lines 1-3; Sec. 6, 3rd Para., 6th Para.)

22. **As to claim 18** (Previously Presented) (incorporating the rejection in claim 15), please refer to above claim **4** accordingly.

23. **As to claim 25** (Previously Presented) (incorporating the rejection in claim 22), please refer to above claim **4** accordingly.

24. **As to claim 31** (Previously Presented) (incorporating the rejection in claim 29), please refer to above claim **4** accordingly.

25. Claims 6, 20, 27, and 33 are rejected under 35 U.S.C. 103(a) as being unpatentable over Lehman in view of Mathur and Victor J. Yodaiken (US Patent 5,995,745) (hereinafter 'Yodaiken' - art made of record)

26. **As to claim 6** (Previously Presented) (incorporating the rejection in claim 1), Lehman discloses the method and the apparatus including means for further specifying *r* preemptive-tasks (e.g., Col. 9, Lines 52-56), *r* being less than or equal to *n*, said task scheduler including a timer mechanism that counts a specified period of time at which time if a preemptive-task is currently executing (e.g., Col. 35, Lines 7-14) and continuing the execution of preemptive-task (e.g., Col. 9, line 64 through Col. 10, line 2)

But Lehman and Mathur do not specifically disclose other limitations stated below.

However, in an analogous art of *Adding Real-Time Support to General Purpose Operating Systems*, Yodaiken discloses the task's state is stored and execution is given to the task scheduler to schedule another task until a later time when the task scheduler restores the state of said preemptive-task. However, in an analogous art, Lake discloses the task's state is stored and execution is given to said task scheduler to schedule another task until a later time when the task

scheduler restores the state of said preemptive-task (e.g., Col. 2, Lines 40-47 - ... RT-executive ... preempting it when needed ...; Col. 4, Lines 49-54 - ... a simple priority-based preemptive scheduler is currently used in RT-Linux ...; Fig. 2, element 64 - ... Restore Interrupt State Return from Interrupt; Col. 5, Lines 24-27 – the low-level "wrapper" routines that save and restore state around calls to handlers have been changed to use soft return from interrupt code ... - emphasis added)

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the teachings of Yodaiken into the Lehman-Mathur system in order to further provide other limitations stated above in the Lehman-Mathur system.

The motivation is that it would further enhance the Lehman-Mathur's system by taking, advancing and/or incorporating the Yodaiken's system which offers significant advantages to operate a real-time operating system, or executive, and retain the capabilities offered by a general purpose operating system as once suggested by Yodaiken (e.g., Col. 1, Lines 60-63)

27. **As to claim 20** (Previously Presented) (incorporating the rejection in claim 15), please refer to above claim **6** accordingly.

28. **As to claim 27** (Previously Presented) (incorporating the rejection in claim 22), please refer to above claim **6** accordingly.

29. **As to claim 33** (Previously Presented) (incorporating the rejection in claim 29), please refer to above claim **6** accordingly.

Conclusion

30. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Ben C. Wang whose telephone number is 571-270-1240. The examiner can normally be reached on Monday - Friday, 8:00 a.m. - 5:00 p.m., EST.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Tuan Q. Dam can be reached on 571-272-3695. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

/Ben C Wang/
Examiner, Art Unit 2192

/Michael J. Yigdall/
Primary Examiner, Art Unit 2192